

EXHIBIT 4

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ARISTA NETWORKS, INC.
Petitioner

v.

CISCO TECHNOLOGY, INC.
Patent Owner

Case No.: IPR2016-00244
Patent 7,953,886

**DECLARATION OF
DOUGLAS W. CLARK**

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

Ex. 1018
IPR of U.S. Pat. No. 7,953,886

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

TABLE OF CONTENTS

I, Douglas W. Clark, hereby declare as follows:	1
I. Qualifications.....	1
II. List of Documents Considered in Formulating My Opinion	3
III. Understanding of Patent Law	4
IV. Person of Ordinary Skill in the Art.....	6
V. Technical Background.....	7
A. The '886 Patent	7
B. The State of the Art	9
1. Command Line Interface (CLI)	9
2. Extensible Markup Language (XML).....	12
3. Use of XML to Configure Network Equipment.....	14
4. Identifying, Extracting, and Arranging Components of Commands Received in an XML Format	16
VI. Claim Construction.....	18
A. “command line interface (CLI) parser”	19
B. “parsing the output message to identify at least one CLI token”	20
VII. Prior Art Forming the Basis for Unpatentability of the '886 Patent	21
A. Courtney	21
B. Gorthy.....	22
C. Froyd.....	24
D. JUNOScript API Guide	25
E. Motivation to Combine	26
VIII. Grounds for Invalidity of the '886 Patent.....	30
A. Claim 1	30
1. [1A] “A method comprising:”	30
2. [1B.1] “receiving, with a command line interface (CLI) parser, an input command configured to request an operation be performed by a routing system,”	30

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

3.	[1B.2] “wherein the input command is configured in an extensible markup language (XML) format having a CLI syntax with CLI keywords sequenced according to configuration rules for CLI commands;”	32
4.	[1C.1] “translating, with the CLI parser, the input command from the XML format having the CLI syntax into a CLI command that, when executed, is configured to prompt the routing system to perform the operation,”	33
5.	[1C.2] “wherein the translating of the input command into the CLI command includes identifying at least one XML tag that includes an XML parameter to indicate the XML tag includes one or more CLI keywords, extracting the one or more CLI keywords from the input command, and arranging the one or more CLI keywords into the CLI command according to the CLI syntax of the input command,”	35
6.	[1C.3] “wherein the routing system is configured to perform the operation responsive to the execution of the CLI command;”	40
7.	[1D] “translating an output message, generated in response to performance of the operation, from a CLI format into an XML format having the CLI syntax, wherein the translating includes parsing the output message to identify at least one CLI token, translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values, and generating the output message in the XML format with the XML values;”	41
8.	[1E] “and transmitting the output message in the XML format having the CLI syntax to a remote device external from the routing system.”	46
B.	Claim 2	48
1.	“The method of claim 1, wherein the input command is formatted in accordance with an XML schema of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem.”	48

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

C.	Claim 3	49
1.	[3A] “The method of claim 2, wherein the translation of the input command from XML format having a CLI syntax into a CLI command comprises:”	49
2.	[3B] “parsing the input command to identify an XML command attribute;”	49
3.	[3C] “traversing the input after the XML command attribute to identify any keywords and any parameters associated with the XML command attribute;”	51
4.	[3D] “translating the XML command attribute into the CLI command;”	51
5.	[3E] “and translating the keywords and any parameters into associated attributes of the CLI command.”	52
D.	Claim 4	53
1.	“The method of claim 1, wherein the output message in the XML format having the CLI syntax includes data formatted in accordance with an XML data model of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem.”	53
E.	Claim 5	54
1.	[5A] “The method of claim 1, wherein the translation of the output message from the CLI format into the XML format having the CLI syntax comprises:”	54
2.	[5B] “parsing the output message to identify at least one CLI token;”	54
3.	[5C] “accessing a stored mapping of CLI tokens-to-XML values;”	55
4.	[5D] “translating each CLI token of the output message into a corresponding XML value, in accordance with said stored mapping;”	56
5.	[5E] “and generating the output message in the XML format having the CLI syntax with the XML values.”	56
F.	Claim 6	58

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

1. [6A] “A computer-usable memory device having computer-readable program code embedded therein for causing a computer system to:”58
 2. [6B] “receive an input command requesting an operation be performed by a routing system, wherein the input command is configured in an extensible markup language (XML) format having command a line interface (CLI) syntax with CLI keywords sequenced according to configuration rules for CLI commands;”59
 3. [6C] “translate the input command from the XML format having the CLI syntax into a CLI command, wherein the routing system is configured to execute the CLI command and perform the operation, and wherein the computer system is further configured to translate the input command by identifying at least one XML tag that includes an XML parameter to indicate the XML tag includes one or more CLI keywords, extracting the one or more CLI keywords from the input command, and arranging the one or more CLI keywords into the CLI command according to the CLI syntax of the input command;”60
 4. [6D] “translate an output message from a CLI format into XML format having the CLI syntax, wherein the output message is generated in the CLI format by the routing system responsive to the performance of the operation, and wherein the translating includes parsing the output message to identify at least one CLI token, translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values, and generating the output message in the XML format with the XML values;”61
 5. [6E] “and transmit the output message in the XML format having the CLI syntax to a remote device external from the routing system.”62
- G. Claim 762
1. “The computer-usable memory device of claim 6, wherein the input command is formatted in accordance with an XML schema of CLI rules and behaviors

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

	enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem.”	62
H.	Claim 8	64
1.	[8A] “The computer-usable memory device of claim 7, wherein the translation of the input command from XML format having a CLI syntax into a CLI command comprises:”	64
2.	[8B] “parsing the input command to identify an XML command attribute;”	64
3.	[8C] “traversing the input after the XML command attribute to identify any keywords and any parameters associated with the XML command attribute;”	66
4.	[8D] “translating the XML command attribute into the CLI command;”	66
5.	[8E] “and translating the keywords and any parameters into associated attributes of the CLI command.”	67
I.	Claim 9	68
1.	“The computer-usable memory device of claim 6, wherein the output message in the XML format having the CLI syntax includes data formatted in accordance with an XML data model of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem.”	68
J.	Claim 10	69
1.	[10A] “The computer-usable memory device of claim 6, wherein the translation of the output message from the CLI format into the XML format having the CLI syntax comprises:”	69
2.	[10B] “parsing the output message to identify at least one CLI token;”	70
3.	[10C] “accessing a stored mapping of CLI tokens-to- XML values;”	70
4.	[10D] “translating each CLI token of the output message into a corresponding XML value, in accordance with said storage mapping;”	71

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

5. [10E] “and generating the output message in the XML
format having the CLI syntax with the XML values.”72

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

I, Douglas W. Clark, hereby declare as follows:

I. Qualifications

1. I received a Ph.D. in computer science from Carnegie-Mellon University in 1976 and a Bachelor of Science in engineering and applied science from Yale University in 1972. Since receiving my doctorate, I have devoted my professional career to the research, design, development, study, and teaching of numerous aspects of computer systems architecture and design. I have studied, taught, practiced, and researched in the field of computer science for over forty years.

2. I am currently a Professor of Computer Science at Princeton University. I have held this position since 1993. I have taught numerous courses at Princeton, including an introductory computer science course, a foundational course for first-year students, and advanced courses in computer architecture for upper-level undergraduates and graduate students. I have also taught a number of advanced graduate seminars about various computer science topics. I have taught most of these courses several times. In addition, in 1990-91, I taught as a Visiting Lecturer in the Division of Applied Sciences at Harvard, and in 2003 I was a Visiting Professor of Computing and Information Science at the University of Pennsylvania.

3. In addition to my experience in academia, I have over 17 years of industrial experience designing computer systems. From 1976 to 1980, I was a Member of the Research Staff at the Xerox Palo Alto Research Center, where I

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

worked chiefly on the design of the Dorado, one of the earliest high-performance workstations. From 1980 to 1993, I worked for the Digital Equipment Corp., first as a Principal Engineer in the Systems Architecture Group, and then as a Consulting and Senior Consulting Engineer in both the Advanced VAX Systems Engineering and Alpha Advanced Development groups. I worked mainly on the architecture, organization, design, simulation, and performance analysis of VAX and Alpha computers. I was one of the principal designers of the VAX 8700 and VAX 8800 – both highly successful machines of the late 1980's.

4. I have authored or co-authored about 60 refereed academic publications in the fields of computer science and engineering. In addition, I have been a referee or associate editor for the following academic journals: ACM Transactions on Computers, IEEE Transactions on Computers, and IEEE Computer. My curriculum vitae includes a list of publications on which I am a named author.

5. I have also been a program committee member or co-chair at a number of national and international conferences/symposiums, including the International Conference on Computer Design, SIGMETRICS Conference on Measurement and Modeling of Computer Systems, and International Symposium on Computer Architecture.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

6. My curriculum vitae, attached herein, contains further details regarding my experience, education, publications, and other qualifications to render an expert opinion in connection with this proceeding.

7. I am being compensated at my normal consulting rate for my work. I receive no other forms of compensation related to this case nor do I have an ownership in either Cisco Systems, Inc. or Arista Networks, Inc. My compensation is not dependent on or otherwise contingent upon the results of this proceeding and in no way affects the substance of my statements in this Declaration.

II. List of Documents Considered in Formulating My Opinion

8. In formulating my opinion, I have considered the following documents:

Exh. No.	Description
1001	U.S. Patent No. 7,953,886 to Bansal et al., issued May 31, 2011 (“886 patent”).
1002	U.S. Patent No. 7,200,548 to Courtney, issued April 3, 2007 (“Courtney”).
1003	U.S. Patent No. 8,296,400 to Gorthy et al., issued October 23, 2012 (“Gorthy”).
1004	U.S. Patent No. 7,155,496 to Froyd et al., issued December 26, 2006 (“Froyd”).
1005	JUNOScript API Guide Release 5.1 (2d ed.), Nov. 6, 2001 (“JUNOScript Guide”).
1006	Honeywell Bull, Multics: Commands and Active Functions (Feb. 1985).
1007	Dennis M. Ritchie & Ken Thompson, The UNIX Time-Sharing System, 17 Communications of the ACM 365 (Jul. 1974).

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

Exh. No.	Description
1008	OpenVMS User's Manual, Version 7.1 (Nov. 1996), http://www.mi.infn.it/~calcolo/OpenVMS/ssb71/6489/6489p.htm .
1009	Brian W. Kernighan & Rob Pike, <i>The UNIX Programming Environment</i> (1984).
1010	Byung-Joon Lee, et al., <i>X-CLI: CLI-Based Management Architecture Using XML</i> (2003) ("Lee").
1011	Alfred V. Aho et al., <i>The AWK Programming Language</i> (1988).
1012	Cisco's Preliminary Claim Construction Disclosure, <i>Cisco Systems, Inc. v. Arista Networks, Inc.</i> , No. 5:14-cv-05344-BLF (N.D. Cal. Aug. 24, 2015).
1013	McGraw-Hill Dictionary of Scientific and Technical Terms, 6th ed. (2003).
1014	Erik T. Ray, <i>Learning XML</i> , 2d ed. (O'Reilly Media, Inc. 2003).
1015	JUNOScript API Guide and Reference Release 4.3 (5d ed.), Nov. 13, 2001.
1016	James Boney, <i>Cisco IOS In a Nutshell</i> , 2d ed. (O'Reilly Media, Inc. 2005).
1017	Dictionary of Computer Science, Engineering, and Technology (2010).

III. Understanding of Patent Law

9. I have been informed and understand that prior art to the '886 patent includes patents and printed publications in the relevant art that predate the date upon which the alleged invention of the alleged invention of the '886 patent was made. I understand that, under 35 U.S.C. § 102(b), a patent is invalid if the invention was described in a patent or printed publication more than one year before the

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

patent at issue was applied for or if, under 35 U.S.C. § 102(e), the invention was described in a U.S. patent application before the patent at issue's invention.

10. I appreciate that my analysis requires an understanding of the scope of the '886 patent claims to a person having ordinary skill in the relevant art at the time the alleged invention was made ("POSA"). Thus, my analysis of the '886 patent and the prior art is made from the perspective of a POSA at the time of the alleged invention of the '886 patent. In addition, I am aware that patent claims subject to *inter partes* review are given the "broadest reasonable interpretation" as would be understood by a POSA.

11. I have been informed and understand that a claim limitation may be expressed as a means for performing a specified function. I understand that for such "means-plus-function" claim limitations, the claim construction of that limitation must include both the function of the claim limitation and the structure corresponding to that function as described in the specification. I further understand that, where the disclosed structure is a general purpose computer, the specification must also disclose an algorithm for performing the function.

12. I further understand that a claim is invalid if it is anticipated or obvious. Anticipation requires that each and every element of a claim be disclosed expressly or inherently in a single prior art reference. I have been informed that a ref-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

erence inherently discloses a feature if the feature in question is necessarily present in what is disclosed by the reference.

13. I understand that a claim may be obvious if common sense directs one to modify prior art, combine multiple examples in a prior art reference, or combine multiple prior art references to add missing features to reproduce each and every element of the alleged invention recited in the patent's claims.

14. I also understand that so-called secondary considerations may be relevant to determine whether a claim is obvious should Patent Owner Cisco Systems, Inc. allege such evidence. Such considerations can include evidence of the invention's commercial success, evidence of a long-felt need that was solved by the invention, evidence that others copied the invention, or evidence that the invention achieved a surprising result. I further understand that there must be a nexus or causal relationship between this evidence and the elements of the claim in order for the consideration to be relevant to the obviousness or non-obviousness of the claim.

IV. Person of Ordinary Skill in the Art

15. I understand that a POSA is one who is presumed to be aware of all pertinent art, thinks along the lines of conventional wisdom in the art, and is a person of ordinary creativity.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

16. In my opinion, a POSA at the time of the alleged invention of the '886 patent would have at least a bachelor's degree in computer science and 3-5 years of experience in systems development.

V. Technical Background

A. The '886 Patent

17. The '886 patent (Ex. 1001), entitled "Method and System of Receiving and Translating CLI Command Data Within a Routing System," was filed on July 8, 2005 and issued on May 31, 2011. The patent is directed to improving the user interface for Cisco routers. Historically, network engineers configured and accessed Cisco routers through the command line interface (CLI) used with Cisco's "internetwork operating system" (known as "IOS"). *Id.* at 1:12-16. As the Background section of the '886 patent explains, however, "IOS CLI is not the most program-friendly of interfaces" *Id.* at 1:26-27. By 2005, "[t]wenty years of consistency and backwards-compatibility" had resulted in a "complicated input and output scheme" that users had to "sort through . . . to input information and extract important data," and these tasks had "proven to be . . . very difficult and cumbersome . . . to automate." *Id.* at 1:26-34.

18. To address this limitation, the '886 patent proposes a "more structured approach to accessing and configuring a router, while still making use of the significant advantages and experience associated with IOS CLI." *Id.* at 1:34-37. This

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

approach entails receiving commands configured in an XML format, rather than the cumbersome CLI format, and using a CLI parser to translate the XML commands into CLI commands that can be processed by the router. Table 1 of the '886 patent provides an example of such an input command in XML format:

TABLE 1

```

<add>
<k_mpls_label>
<k_range>
<range-min>10</range-min>
  <range-max>300</range-max>
<k_static>
  <static-min>30</static-min>
  <static-max>150</static-max>
</k_static>
</k_range>
</k_mpls_label>
</add>

```

Ex. 1001 at Table 1.

19. The CLI command corresponding to this example is “mpls label range 10 300 static 30 150.” *Id.* at 5:57-65. The XML tags shown in Table 1—which are delimited by angle brackets (“<” and “>”)—contain “CLI keywords” or enclose “command parameters,” both of which the parser extracts and arranges into CLI commands. *Id.* at 5:5-6 and 5:30-36. In this example, “mpls label,” “range,” and “static” are CLI keywords, each contained in the tags with the corresponding text; “10” and “300” are command parameters enclosed respectively by the tags “range-min” and “range-max”; and “30” and “150” are command parameters enclosed respectively by the tags “static-min” and “static-max.”

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

20. The '886 patent further describes that, when an operation invoked by a CLI command generates an output message (i.e., the text received in response to sending a command to the router), the output message is translated from a "CLI format into an XML format having the CLI syntax." *Id.* at 7:59-61. To perform this translation, the system parses the output message "to identify at least one CLI token," and then "translat[es] each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values." *Id.* at 7:62-66. The translated output message, now in XML format, is then transmitted to a remote device external from the routing system, such as the computer that sent the input command. *Id.* at 8:1-3.

B. The State of the Art

1. Command Line Interface (CLI)

21. A command line interface (often abbreviated as "CLI") is a type of user interface to a computer system in which the user types commands in response to a prompt and receives responses from the system on subsequent lines. Command line interfaces have long been well-known in the art. For example, the influential MULTICS system, developed at MIT in the 1960's, had a set of programs, each invoked by a command: "A command performs some action for you, such as displaying information on your terminal, formatting a report, or compiling a program." Honeywell Bull, Multics: Commands and Active Functions (Feb. 1985)

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) (Ex. 1006) at 21. The even more influential Unix operating system, developed at Bell Labs in the 1970's, had a CLI embodied in its "Shell:" "The Shell is a command line interpreter: it reads lines typed by the user and interprets them as requests to execute other programs. In simplest form, a command line consists of the command name followed by arguments to the command" D. Ritchie & K. Thompson, *The UNIX Time-Sharing System*, 17 *Communications of the ACM* 365 (Jul. 1974) (Ex. 1007) at 7. A third example is the Digital Equipment Corporation's Digital Command Language or DCL, developed in the 1980's or earlier, which was "a set of English-like instructions that tell the operating system to perform specific operations." *OpenVMS User's Manual, Version 7.1* (Nov. 1996) (Ex. 1008) at 17, Section 3.2. The manual explains that "[l]ike a spoken language, DCL is made up of *words* (vocabulary) and *word order* (syntax or format)," and that "[j]ust as a spoken language depends on the order of words to create meaning, DCL requires that you put the correct elements of the command line in a specific word order or format." *Id.* at 18, Section 3.3 and 19, Section 3.3.3.

22. CLIs are text-based interfaces and generally operate in a similar manner. A user is presented with a command prompt (e.g., ">" or "C:\>"), after which the user may type a command requesting that the system perform a particular operation. The user must generally know what commands are valid for the system in question, as well as the required syntax for parameters associated with those com-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

mands. For example, in UNIX, a user can type the command “ls -l ” to cause the operating system to show a long-form list of the contents of the current directory:

The -l option [of the ls command] gives a “long” listing that provides more information about each file:

```
$ ls -l
total 2
-rw-r--r--  1 you          19 Sep 26 16.25 junk
-rw-r--r--  1 you          22 Sep 26 16.26 temp
$
```

B. Kernighan & R. Pike, *The UNIX Programming Environment* (1984) (Ex. 1009) at 19. As shown in the above example, in response to the command, the computer system will typically return some text that constitutes the response to the command, or some other text to indicate that an error occurred (perhaps, for example, a misspelling or mis-formatting of the command). The user is then returned to the command prompt (the dollar sign in the Unix example above), where another command can then be entered.

23. Cisco’s Internetwork Operating System (IOS) had been in use for over twenty years when the ’886 patent was applied for. *See* Ex. 1001 at 1:27-30. It uses a CLI that is similar to ones used in the prior art. IOS software runs on Cisco routers, which are devices that enable packetized information to be passed from one machine, through one or more routers, to a destination machine. James Boney, *Cisco IOS In a Nutshell*, 2d ed. (O’Reilly Media, Inc. 2005) (Ex. 1016) at 9. The routers are accessed and configured by administrators via a serial port, ssh, or other

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

connection from a remote computer. *See id.* at 16-17, 32. Using the CLI, the administrator can access and configure the router by entering textual commands that follow the proper syntax. *See id.* at 9. Just as with prior art CLIs, in response to an entered command, the router may respond with requested information, make some change in the router's settings, or, in the event of an incorrect command, report an error.

2. Extensible Markup Language (XML)

24. Extensible markup language, or “XML,” is an open standard markup language that was developed in the 1990s. Erik T. Ray, *Learning XML*, 2d ed. (O'Reilly Media, Inc. 2003) (Ex. 1014) at 16. A “markup language” is a language that allows users to annotate digital documents, usually by embedding the annotations in the document's text. These annotations can be used for many purposes. For example, they may instruct a web browser to display certain text in a particular way, or to make a particular string of text a hypertext link. Annotations may also describe portions of a document—for example, they may label certain text as a street name within an address, or as a citation, and software may then use this information in a variety of ways. *Id.* at 24-26. Hypertext markup language, or HTML, is one well-known markup language that is used to create web pages.

25. Like other markup languages, XML features “tags” embedded within a document. *Id.* at 57. Tags are delimited by angle brackets (“<” and “>”). *Id.* at

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

11. With certain exceptions, tags include a start tag and an end tag (in which the name of the tag is preceded by a backslash). An XML “element” consists of a start tag, a corresponding end tag, and everything in between (except in the case of an “empty element,” which I discuss below). *Id.* at 58. For example, an XML element might consist of:

```
<street address>2800 S. Randolph Street</street address>
```

XML tags may also contain “attributes” that describe the element. *Id.* at 67. For example, an XML element might consist of:

```
<name language=“french”>Pierre-Auguste</name>
```

In this example, “language” is the name of the attribute and “french” is the attribute’s value. Elements may also be nested within other elements. *Id.* at 58. For example, an XML element might consist of:

```
<name language=“french”>
  <first name>Pierre-Auguste</first name>
  <last name>Renoir</last name>
</name>
```

In this example, the “first name” element and “last name” element are nested within the “name” element. As noted above, XML also allows “empty elements,” which are elements with start tags only, containing no nested elements or other contents, and whose start tags are terminated by a slash character. Empty elements have attributes within their start tags. *Id.* at 66. For example, an XML element may consist of:

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

<artist name="Pierre-Auguste Renoir"/>

The slash at the end of the tag in this example indicates that it has no matching end tag. The tag itself constitutes the XML element. *Id.*

26. XML is an “extensible” markup language because it allows users to define their own systems of tags and attributes. *Id.* at 10-11. For any particular use of XML, the tags that can be used in an XML document, the type of data they represent, and their hierarchical relationships may be defined in what is known as an “XML schema.” *See id.* at 87-88. A schema acts as a blueprint of the possible combinations of tags and attributes that may be used in an XML document that conforms to a particular application’s syntax and formats. *See id.* at 87-88, 90, 92. Schemas were used to model data structures in the prior art long before the ’886 patent. *Id.* at 87-88.

3. Use of XML to Configure Network Equipment

27. Not long after XML emerged as a versatile and adaptable open standard for data description and transmission, emerging network- technology companies realized its potential use for router configuration. By January 2001—over four years before the ’886 patent was applied for—Cisco’s competitor Juniper Networks, Inc. had released an XML-based interface (JUNOScript) for use with its routers, allowing administrators to access and configure the routers using XML commands instead of CLI. *See* JUNOScript API Guide and Reference Release 4.3

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) (5d ed.), Nov. 13, 2001 (Ex. 1015) at 2 (indicating first edition date of January 22, 2001). The JUNOScript API provided a set of XML tags (which it calls JUNOScript tags) that client applications could send to the router. *Id.* at 45. Software in the router would process the request, encode the response to the request in JUNOScript tags, and return the result to the client. *Id.* The client could thus transmit an XML command to the router and receive a response formatted in XML. *See id.*; *see also* JUNOScript API Guide Release 5.1 (2d ed.), Nov. 6, 2001 (“JUNOScript Guide”) (Ex. 1005).

28. The shift toward XML, as opposed to CLI, to configure and communicate with network equipment was praised in the industry long before the '886 patent was filed. For example, also in 2001, network- software company Intelliden filed patent applications praising the “user-friendly XML-based interface” of the Juniper Networks routers, while noting that “Cisco™ routers are notoriously difficult to configure” due to their “cumbersome command line interface (CLI).” U.S. Patent No. 8,296,400 (“Gorthy”), issued October 23, 2012 (Ex. 1003) at 1:47-51; *see also* U.S. Patent No. 7,200,548 (“Courtney”), issued April 3, 2007 (Ex. 1002) at 1:47-52. Intelliden acknowledged Cisco’s bind: “If Cisco™ attempted to abandon its CLI in favor of the new user-friendly XML-based interface, many years of development and expertise could be lost.” Ex. 1003 at 1:56-58; *see also* Ex. 1002 at 1:61-63. Anticipating the solution described by the '886 patent almost four years

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) later, Intelliden filed a patent application that would issue as both the Gorthy patent and the Courtney patent, disclosing a means for converting XML-based commands to CLI commands, and converting CLI output to XML, for use with Cisco routers. *See, e.g.*, Ex. 1003 at 6:7-12, 6:52-59; Ex. 1002 at 2:15-18, 2:32-37.

29. Others were also developing XML interfaces for existing CLI systems around this time. For example, in March 2003, Lee and co-authors published *X-CLI: CLI-Based Management Architecture Using XML* (Ex. 1010), which discloses an API for configuring CLI-based network systems using XML. *Id.* at 3. Lee describes an XML template representing a group of CLI commands in a hierarchical manner that is converted to “actual CLI commands, and sent to the network devices by the X-CLI API interfaces.” *Id.* In addition, U.S. Patent No. 7,155,496 (“Froyd”) (Ex. 1004), filed January 29, 2002, discloses a framework for delivering configuration data in an XML format to and from a system. *Id.* at Abstract. The XML-formatted data, which includes tags containing CLI keywords, is converted into CLI commands that are executed by the system. *Id.* at 8:51-67, 12:44-48.

4. Identifying, Extracting, and Arranging Components of Commands Received in an XML Format

30. A person of ordinary skill in the art at the time of the ’886 patent application would have understood that a CLI command in XML format could readily be transformed by using, for example, the well-known text-manipulation tools of Unix, such as “sed,” “grep,” and “awk.” Ex. 1009 at Chapter 4. Identifying and ex-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

tracting a CLI keyword from an XML tag (or a CLI command parameter enclosed by XML tags) and then arranging the results in the order in which they appear in the XML-formatted command (in other words, in CLI syntax) is a straightforward application of these tools. A simple example from *The AWK Programming Language*, published in 1988, demonstrates identifying, extracting and arranging using the awk tool. It starts with an example file “that contains the name, pay rate in dollars per hour, and number of hours worked for your employees:”

Beth	4.00	0
Dan	3.75	0
Kathy	4.00	10
Mark	5.00	20
Mary	5.50	22
Susie	4.25	18

Alfred V. Aho et al., *The AWK Programming Language* (1988) (Ex. 1011) at 9.

The text explains that, using an awk command, “[y]ou can also print words in the midst of fields and computed values:

```
{ print "total pay for", $1, "is", $2 * $3 }
```

prints

```
total pay for Beth is 0
total pay for Dan is 0
total pay for Kathy is 40
total pay for Mark is 100
total pay for Mary is 121
total pay for Susie is 76.5
```

Id. at 15.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

In this example it is clear that the employee records have been *identified*, the employee names, wage, and other information have been *extracted* from the original file, and then the names, wage, and hour information have been *arranged* in new lines in the output. A person of ordinary skill would further have understood that the “printed” result of an awk command could be “re-directed” to a file for further use. *Id.* at 63.

31. A person of skill in the art would have understood that these and other techniques for identifying, extracting, and arranging could readily be applied to XML. Indeed, transforming XML into a different form was a well-known use for XML long before the ’886 patent was filed. As the book *Learning XML* explained in 2003, “[t]ransformation requires two things: the source document and a transformation stylesheet. The stylesheet is a recipe for how to ‘cook’ the XML and arrive at a desired result. . . . [Transformation] can be used to turn an XML document into just about any form you can imagine.” Ex. 1014 at 50-51.

VI. Claim Construction

32. As noted above, I understand that the challenged claims must be given their broadest reasonable interpretation in light of the specification of the ’886 patent, which means that the words of the claims should be given the broadest possible meaning, as understood by a person of ordinary skill in the art, consistent with

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

the patent's specification. Below, I offer my opinion as to the broadest reasonable interpretation for some claim elements of the '886 patent.

A. “command line interface (CLI) parser”

33. It is my opinion that the broadest reasonable interpretation of “command line interface (CLI) parser” in the '886 patent is “a subsystem of the routing system capable of receiving input commands, translating those commands into CLI commands, and parsing the received and translated CLI commands.”

34. The claims of the '886 patent recite that the CLI parser receives commands in an XML format and translates them to a CLI format. For example, they state that the “command line interface (CLI) parser” “receiv[es] . . . an input command” configured in an XML format having a CLI syntax, and “tranlat[es] the input command from the XML format having the CLI syntax into a CLI command” Ex. 1001 at 7:38-47; *see also id.* at Fig. 1 (showing IOS/CLI Parser 110).

35. Similarly, the specification explains that command statements are “received by routing system 100 through programming port 103, and are passed to IOS/CLI parser 110.” *Id.* at 3:20-22. The received command statements can be in a CLI format or an XML format—that is, they can be “formatted in accordance with the CLI rules and behaviors expected by IOS/CLI Parser 110, or in accordance with an XML schema of the CLI rules and behaviors.” *Id.* at 3:23-26. “Non-CLI

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) command statements” (which would include input commands in an XML format) “must . . . be translated into CLI statements that Command Module 130 can interpret.” *Id.* at 3:34-36. The “command line interface (CLI) parser” performs this translation, as shown in Figure 2, which is “a flowchart of a method for *receiving* and *translating* data using an internetwork operating system (IOS) *command line interface (CLI) parser subsystem* of a routing system, in accordance with one embodiment of the present invention.” *Id.* at 1:44-48 (emphasis added). Once the CLI commands are received by the CLI parser, either directly or after being translated from an XML format, “the command statements are parsed according to instructions programmed into IOS/CLI Parser 110.” *Id.* at 3:29-31.

B. “parsing the output message to identify at least one CLI token”

36. In the related litigation, Cisco has proposed that the term “parsing the output message to identify at least one CLI token” be construed as follows: “analyzing the output message to extract at least one unit of CLI characters in a sequence.” Cisco’s Preliminary Claim Construction Disclosure, Cisco Systems, Inc. v. Arista Networks, Inc., No. 5:14-cv-05344-BLF (N.D. Cal. Aug. 24, 2015) (Ex. 1012) at 26. I agree that Cisco’s construction would be the broadest reasonable interpretation of this term. The term “parsing” generally refers to a “process by which an input string is analyzed using a grammar to determine if the input string satisfies the rules of the grammar.” Dictionary of Computer Science, Engineering,

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) and Technology (2010) (Ex. 1017) at 4; *see also* Ex. 1012 at 26. The term “token” generally refers to “[a] distinguishable unit in a sequence of characters.” McGraw-Hill Dictionary of Scientific and Technical Terms, 6th ed. (2003) (Ex. 1013) at 6 (definition 1).

VII. Prior Art Forming the Basis for Unpatentability of the ’886 Patent

A. Courtney

37. U.S. Patent No. 7,200,548 to Courtney, entitled “System and Method for Modeling a Network Device’s Configuration,” issued on April 3, 2007. The application that matured into Courtney was filed on August 29, 2001. Courtney is accordingly prior art under 35 U.S.C. § 102(e). Moreover, the Courtney application was published on March 6, 2003, making the published application prior art under 35 U.S.C. § 102(b).

38. Courtney discloses a “system and method for modeling the configuration of a network device” that may include “a CLI-to-XML converter.” Ex. 1002 at 2:32-37. As with the ’886 patent, the inventors explained that one reason for using the Courtney invention would be to “model a network device’s configuration” by “converting it into a standard-format configuration such as an XML document or a DOM.” *Id.* at 2:40-45. This was beneficial to users because “instead of being forced to manipulate a difficult CLI-based configuration format, or other format system administrators can use the standard-format configuration to interact with

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

the target network device.” *Id.* at 2:48-51. In order to perform the translation to XML, the system could “generate[] an XML representation of each native-format command in the network device’s configuration by associating each command with the schema, or its hash representation.” *Id.* at 3:23-27; *see also id.* at 6:50-53 (“[T]he XML converter 235, using the appropriate schema, generates an XML document containing an XML representation of the network device’s configuration.”).

39. The Courtney system uses schema information to take the native-format configuration and “assemble the XML-based command and write it to the corresponding XML document.” *Id.* at 7:31-36. That process is repeated for each command in the device’s native-format configuration until “all native-format commands have been converted.” *Id.* at 7:37-48.

B. Gorthy

40. U.S. Patent No. 8,296,400 to Gorthy *et al.*, entitled “System and Method for Generating a Configuration Schema,” issued on October 23, 2012. The application that matured into Gorthy was filed on August 29, 2001. Gorthy is accordingly prior art under 35 U.S.C. § 102(e). Moreover, the Gorthy application was published on March 13, 2003, making the published application prior art under 35 U.S.C. § 102(b).

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

41. Gorthy discloses the creation of a configuration schema generated from a collection of all of the commands found in a Cisco router. Ex. 1003 at 2:30-59. That configuration schema can then “be used to generate commands.” *Id.* at 2:63-65; *see also id.* at 3:9-11 (“[T]he schema can be used to generate CLI commands from, for example, XML-based commands.”). Thus, “[w]hen given a command in XML format, the command information in the configuration schema can be used to reformat the XML-based command into a proper CLI format.” *Id.* at 3:13-16. After reformatting, the command can be sent to the router, allowing a system administrator to configure routers without knowing the details of the CLI for the router. *See id.* at 3:16-19.

42. Specifically, in operation, the invention of Gorthy allows an “XML-based command [to] be passed to the converter 235 which converts the XML-based command to a CLI-based command using the XML schema.” *Id.* at 6:9-11. Thereafter, the “CLI-based command, not the XML command, can then be passed to the configuration storage module 145 where it is integrated into the configuration of the router.” *Id.* at 6:11-14. The device that converts XML commands to CLI commands need not be a part of the router itself; it can be based with the system administrator or elsewhere in the network. *See id.* at 6:16-40.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

C. Froyd

43. U.S. Patent No. 7,155,496 to Froyd, (Froyd”) (Ex. 1004) entitled “Configuration Management Utilizing Generalized Markup Language,” issued on December 26, 2006. The application that matured into Froyd was filed on January 29, 2002, as a continuation-in-part of an application filed in May of 2001. Froyd is accordingly prior art under 35 U.S.C. § 102(e). Moreover, the Froyd application was published on June 19, 2003, making the published application prior art under 35 U.S.C. § 102(b).

44. Froyd discloses a framework for delivering data to and from a system. It recognizes that data in an XML format is oftentimes more versatile for users than data kept in a native format such as that used by CLI. Ex. 1004 at 15:57-63 (“The statistic information is returned to the user in the XML format. This allows the statistic information to be displayed and presented to the user through the browser. Placing the statistic information into the XML format also allows the statistic information to be used by other devices capable of processing data in the XML format.”).

45. Accordingly, Froyd proposes a system for sending and receiving information, including specifically information about how a system is or should be configured, that converts the information between XML and CLI formats. “The configuration data may then be saved in a configuration file 818 for a subsequent

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

restore operation. In one embodiment, the configuration file is saved in XML format. When a restore function is invoked, the configuration file 818 is processed into CLI command lines. The CLI command lines are fed back to the CLI 805.” *Id.* at 12:42-48.

46. The Froyd system also recognizes the need to make the commands that it uses consistent with the language used by the router in question. “XML can be used to create a new set of commands that describe commands associated with a Lucent router such that the new set of commands is consistent with commands associated with routers from Cisco.” *Id.* at 11:20-24.

D. JUNOScript API Guide

47. JUNOScript API Guide Release 5.1 (2d ed.) (“JUNOScript Guide”) (Ex. 1005), was published on November 6, 2001, by Juniper Networks, Inc. The JUNOScript Guide is accordingly prior art under 35 U.S.C. § 102(b).

48. The JUNOScript Guide describes the use of the JUNOScript application programming interface (API) to configure or request information from software running on Juniper Networks routers. Ex. 1005 at 9. The JUNOScript API uses XML tags that describe router components to perform these operations. *Id.* The XML tags correspond to command line interface (CLI) statements used to configure or request information from the router. *Id.* at 32 (“Configuration requests correspond to the JUNOS CLI configuration statements described in each of the

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

JUNOS Internet software configuration guides. The JUNOScript API defines a tag for every container and leaf statement in the JUNOS configuration hierarchy.”).

49. The JUNOScript Guide describes the benefit of formatting inputs to and outputs from the router in XML tags, noting that the tags “make it straightforward for client applications that request information from a router to parse the output and find specific information.” *Id.* at 17. The Guide illustrates this by showing how the tokens of the ASCII output message received in response to a CLI command are translated into JUNOScript XML values. *Id.*; *see also id.* at 37 (“To display the output from a JUNOS CLI command as JUNOScript tags rather than the default formatted ASCII, pipe the command to the **display xml** command.”)

E. Motivation to Combine

50. One of skill in the art would have every motivation to combine Courtney and Gorthy—indeed, it may not even be necessary to combine the two of them, as Courtney expressly incorporates by reference the application that matured into Gorthy. Ex. 1002 at 3:7-11 (incorporating by reference U.S. Patent Application No. 09/942,834). The applications that matured into Gorthy and Courtney were filed the same day; indeed, they were filed so close together in time that their application serial numbers are consecutive (the application that matured into Courtney is Serial No. 09/942,833, while the application that matured into Gorthy is Serial No. 09/942,834). The two patents describe two facets of a system from a

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

company known as Intelliden (Intelliden was acquired by IBM in 2010, before Gorthy issued, thus explaining the different assignees shown on the face of the two patents).

51. One of ordinary skill in the art would also be motivated to combine Courtney and Gorthy because their inventions are complementary to one another. Courtney explains how to use the conversion of CLI to XML in order to allow configuration information sent from a router to be easily understood. Gorthy provides information about the use of the flip side of this process—how to use the conversion of XML to CLI to allow routers to be configured in a simplified manner.

52. Likewise, one of skill in the art would have been motivated to combine Froyd with either or both Courtney and Gorthy because each is concerned with the use of translations to and from XML in order to provide users with an easy to use standard format. *See, e.g.*, Ex. 1002 at 2:48-51 (“That is, instead of being forced to manipulate a difficult CLI-based configuration format, or other format system administrators can use the standard-format configuration to interact with the target network device.”); Ex. 1003 at 3:5-8 (“Once the CLI-based command has been converted to an XML format, the XML format of the command can be easily passed between various computers and system administrators in a highly readable, standardized format.”); Ex. 1004 at 15:57-63 (“The statistic information is returned to the user in the XML format. This allows the statistic information to

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

be displayed and presented to the user through the browser. Placing the statistic information into the XML format also allows the statistic information to be used by other devices capable of processing data in the XML format.”).

53. Moreover, all three of the references expressly discuss the potential for use with Cisco routers specifically. *See, e.g.*, Ex. 1002 at 2:15-18 (“Accordingly, a system and method are needed that will allow manufacturers, like Cisco™, to create user-friendly interfaces for both next-generation and existing devices.”); Ex. 1003 at 4:39-41 (“The illustrated method can be used, for example, to generate an XML schema from the CLI commands associated with a Cisco™ router.”); Ex. 1004 at 11:20-24 (“XML can be used to create a new set of commands that describe commands associated with a Lucent router such that the new set of command is consistent with commands associated with routers from Cisco.”).

54. One of ordinary skill in the art would further have been motivated to combine the JUNOScript API Guide with Gorthy, Courtney, and Froyd because each reference relates to using XML-tagged commands to configure and request information from routers. Both Gorthy and Courtney discuss the XML-based interface of Juniper Networks routers (Ex. 1003 at 1:47-51; Ex. 1002 at 1:48-51), and Gorthy specifically illustrates that configuration schemas according to the invention could be generated for Juniper Networks routers (*see* Ex. 1003 at Fig. 4). Those references’ praise of such XML-based interfaces would have directed a per-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

son of ordinary skill in the art to documentation regarding Juniper Networks routers, such as the JUNOScript API Guide. Moreover, Froyd and the JUNOScript Guide are directed to the same field of endeavor—frameworks for configuring networking equipment using extensible markup language—and teach similar methods of using XML-based commands with a CLI syntax. *See, e.g.*, Ex. 1004 at Abstract, Fig. 7; Ex. 1005 at 15-16.

55. One of ordinary skill in the art would further be motivated to combine the teachings of the JUNOScript Guide with Gorthy, Courtney, and Froyd because the inventions complement each other. Courtney, which incorporates Gorthy by reference, discloses a CLI-to-XML converter for use with a router (Ex. 1002 at Abstract), and the JUNOScript Guide teaches outputting the results of a CLI command to XML-formatted tags (*see, e.g.*, Ex. 1005 at 37). One of ordinary skill in the art would understand that the JUNOScript Guide discloses possible implementations of CLI-to-XML conversion that could be used consistent with the system disclosed in Gorthy and Courtney. Likewise, Froyd discloses an XML tagging convention for CLI commands that one of ordinary skill in the art would recognize could be used in conjunction with the system of Gorthy/Courtney and the JUNOScript Guide.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

VIII. Grounds for Invalidity of the '886 Patent

56. I understand that the asserted grounds for the unpatentability of the '886 patent is that claims 1-10 are rendered obvious by Gorthy in light of Courtney, Froyd, and the JUNOScript Guide. Below I discuss how each of the claim elements in the '886 patent are rendered obvious.

A. Claim 1

1. [1A] "A method comprising:"

57. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses a method with the steps set forth in the elements below. *See* claim elements 1B-1E, below.

2. [1B.1] "receiving, with a command line interface (CLI) parser, an input command configured to request an operation be performed by a routing system,"

58. Gorthy discloses receiving, with a command line interface (CLI) parser, an input command configured to request an operation be performed by a routing system. For example, Gorthy discloses that the "converter 235" initially receives an XML-based configuration command." Ex. 1003 at 6:43-44; *see also id.* at 6:9-11 ("That XML-based command can be passed to the converter 235 which converts the XML-based command to a CLI-based command using the XML schema.").

59. The XML-based input commands disclosed by Gorthy are configured to request an operation be performed by a routing system. Gorthy indicates that "a

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

system administrator 125 can reconfigure such a router using XML-based commands” *Id.* at 6:4-5. The exemplary schema of Appendix B, which is used to generate certain of the exemplary CLI commands in Appendix A, for example, defines certain “service” configuration commands that would request an operation be performed by a routing system. Moreover, Gorthy explains that “[o]nce reformat- ted into a CLI format, the command can be pushed out to the appropriate router. Thus, a system administrator could configure such a router without knowing the specifics of the CLI.” *Id.* at 3:16-19; *see also id.* at 4:33-35 (“[N]ew configuration commands are provided to the router 120 through the CLI.”). One of skill in the art would understand that a command that is “pushed out to the appropriate router” is configured to request an operation be performed by a routing system, for example, enabling the exemplary “service” features disclosed in Appendices A and B.

60. The above-described input command is received with a “command line interface (CLI) parser.” Gorthy discloses that its routing system includes a converter subsystem. *Id.* at Fig. 5; *see also id.* at 3:41-42, 5:64-67. Gorthy explains that the converter 235 receives XML-based configuration commands and “converts the XML-based command to a CLI-based command” *Id.* at 6:9-11; *see also id.* at 6:43-44. After translating the input command to a CLI command, the con- verter “verifies the correctness and validity of that command, and provides it to the router 120” *Id.* at 6:48-51. Because it receives an XML input command, trans-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

lates the XML input command to a CLI command, and parses the resulting CLI command, the converter of Gorthy discloses the claimed “command line interface (CLI) parser.”

3. [1B.2] “wherein the input command is configured in an extensible markup language (XML) format having a CLI syntax with CLI keywords sequenced according to configuration rules for CLI commands;”

61. Gorthy discloses that the input command is configured in an extensible markup language (XML) format having a CLI syntax with CLI keywords sequenced according to configuration rules for CLI commands. Gorthy explains that the described XML configuration command (input command) is generated using an XML configuration schema. Ex. 1003 at 3:9-11. The XML configuration schema is generated “from the CLI commands associated with a Cisco™ router,” and it “contains the commands, command hierarchy and bounds of the various configuration commands” of the router. *Id.* at 3:11-13, 4:40-41; *see also id.* at 2:59-62. It is “in essence . . . a modeling of the router’s command structure.” *Id.* at 2:63-65, 3:9-11, 5:8-9. The XML configuration schema, and XML-based commands generated from that schema as described in Gorthy, thus have a “CLI syntax with CLI keywords sequenced according to configuration rules for CLI commands.” *See id.* at 3:13-16 (“When given a command in XML format, the command information in the configuration schema can be used to reformat the XML-based command into a proper CLI format.”).

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

62. In Appendix B, Gorthy provides an example of an XML configuration schema corresponding to a portion of one of the CLI commands (the “service” command) in Appendix A. *See id.* at 5:31-34; *see also id.* at Appendix A and B. As reflected in Appendix B, the schema that controls whether a particular XML-based command will be accepted as valid requires that the particular command words be entered in the order in which they are found in valid CLI commands—in other words, that valid input commands must follow CLI syntax. Consider, for example, the following:

```
<service>
  <alignment>
    <detection/>
  </alignment>
</service>
```

63. Appendix B would indicate to one of ordinary skill in the art that this XML-based command is a valid command because its XML tags are named consistent with, and according to the hierarchical relationships defined by, the schema. *See id.* at Appendix B (The “<xsd:element name=‘detection’>” tag is nested within the “<xsd:element name=‘alignment’>” tag, which is nested within the “<xsd:element name =‘service’>” tag.) Such an input command is thus in XML format having a CLI syntax with CLI keywords sequenced according to configuration rules for CLI commands.

4. [1C.1] “translating, with the CLI parser, the input command from the XML format having the CLI syntax into a

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

CLI command that, when executed, is configured to prompt the routing system to perform the operation,”

64. Gorthy discloses translating, with the CLI parser, the input command from the XML format having the CLI syntax into a CLI command that, when executed, is configured to prompt the routing system to perform the operation. As discussed above, Gorthy discloses translating XML-based configuration commands having CLI syntax to CLI-based commands using an XML schema. Ex. 1003 at 3:9-16 (“[T]he schema can be used to generate CLI commands from, for example, XML-based commands. . . . When given a command in XML format, the command information in the configuration schema can be used to reformat the XML-based command into a proper CLI format.”); 6:9-11 (“That XML-based command can be passed to the converter 235 which converts the XML-based command to a CLI-based command using the XML schema.”); *see also* 6:35-40, 6:43-51. A person of ordinary skill would understand that the XML “service” command above would be translated into the CLI command “service alignment detection.” *Id.* at Appendix B, Appendix A.

65. Gorthy also discloses that the converted CLI command, when executed, is configured to prompt the routing system to perform the operation responsive to the CLI command. Gorthy explains, for instance, that the converted CLI configuration command is transmitted through the network to the router where it is used to configure the router. Ex. 1003 at 3:16-19; *see also id.* at 6:9-14 (“Th[e] XML-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

based command can be passed to the converter 235 which converts the XML-based command to a CLI-based command using the XML schema. A CLI-based command, not the XML command, can then be passed to the configuration storage module 145 where it is integrated into the configuration of the router.”); *id.* at 6:21-26 (“the localized converter 235’ and schema storage module 240’ convert the XML-based command to a CLI-based command and then transmit the CLI-based command through the network 250 to the router 120”); *id.* at Fig. 8; *see also* claim element [1B]; Ex. 1003 at 4:33-35.

5. **[1C.2] “wherein the translating of the input command into the CLI command includes identifying at least one XML tag that includes an XML parameter to indicate the XML tag includes one or more CLI keywords, extracting the one or more CLI keywords from the input command, and arranging the one or more CLI keywords into the CLI command according to the CLI syntax of the input command,”**

66. Gorthy in view of Froyd discloses that the translating of the input command into the CLI command includes identifying at least one XML tag that includes an XML parameter to indicate the XML tag includes one or more CLI keywords, extracting the one or more CLI keywords from the input command, and arranging the one or more CLI keywords into the CLI command according to the CLI syntax of the input command.

67. As discussed above, Gorthy discloses translating an XML-based command to a CLI command using an XML configuration schema that models the

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

command hierarchy of a given router. The exemplary XML configuration schema disclosed by Gorthy in Appendix B specifies a series of element name parameters (e.g., “service,” “alignment,” “detection,” “logging”). When viewed in comparison to Appendix A, which discloses exemplary CLI commands for a given router, one of ordinary skill in the art would recognize that each of the element name parameters of Appendix B corresponds with a CLI keyword used in one or more CLI commands. Moreover, the element names in the XML schema are arranged in a hierarchical order that reflects the permissible sequencing of CLI keywords for a given CLI command. For example, nested beneath the “<xsd:element name=’service’>” tag is the “<xsd:element name=’alignment’>” tag. Ex. 1003 at Appendix B. One level below that tag are the “<xsd:element name=’detection’>” and “<xsd:element name=’logging’>” tags. *Id.* The initial CLI commands listed in Appendix A reflect this same hierarchical sequencing: “service,” “service alignment,” “service alignment detection,” and “service alignment logging.” *See id.* at Appendix A.

68. From the XML configuration schema of Appendix B, one of ordinary skill in the art would understand what a valid XML command according to that schema would look like. As discussed above, one example would be:

```
<service>
  <alignment>
    <detection/>
```

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

```

    </alignment>
  </service>

```

See discussion of claim 1B.2, above. As this example illustrates, this XML input command—which would translate to the example command “service alignment detection” in Appendix A—consists of a series of XML tags. Within each of the XML tags is a CLI keyword. Because the XML configuration schema includes element name parameters *only* for CLI keywords, the CLI keywords in the XML tags are themselves parameters that indicate the tag includes a CLI keyword.

69. Of course, one of ordinary skill in the art would recognize that alternative XML schemas could be used with the system disclosed by Gorthy, including schemas that permit XML tags to identify CLI keywords using XML parameters. Froyd discloses XML commands according to such a schema. For example, Froyd teaches that the following XML command could be used:

```

<command>
  <keyword text=“show”>
  </keyword>
</command>

```

Ex. 1004 at 8:58-61. As Froyd explains, “[t]he <keyword> tag 420 specifies the text of a command keyword in the command set of the CLI such as, for example, SHOW, ENABLE, or LOGOUT. The text is supplied by the single attribute ‘text’.” *Id.* at 8:63-66. Here, the XML tag <keyword . . .> includes an XML parameter (“text=”) that indicates the tag includes a CLI keyword (“show”).

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

70. As the above examples of XML commands from Gorthy and Froyd illustrate, there are numerous ways to represent CLI keywords in XML. One of ordinary skill in the art would understand that the decision of how to represent CLI keywords in XML is simply an implementation choice. As these references illustrate, there was nothing novel in 2005—nearly four years after Froyd and Gorthy’s applications were filed—about using some sort of parameter to indicate that an XML tag includes a CLI keyword (as in Froyd), as opposed to an XML tag that contains only the CLI keyword (as in Gorthy). Recognizing that using an XML parameter to indicate the presence of one or more CLI keywords was a non-inventive implementation decision, one of ordinary skill in the art would have found it obvious to use the XML parameter disclosed in Froyd with the XML-to-CLI translation system of Gorthy.

71. The similar objectives of the inventions of Gorthy and Froyd further support such a combination. Froyd, like Gorthy, discloses a system that enables translating XML commands to CLI commands (*see, e.g.*, Ex. 1004 at 17:57-60), and one of ordinary skill in the art would recognize that this common objective would make each reference suitable for improvements from the other.

72. It would have been obvious to one of ordinary skill that translating the XML command to a CLI command, as disclosed in Gorthy, could occur by identifying an XML tag that includes an XML parameter to indicate that the XML tag

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

includes one or more CLI keywords. As discussed above, the XML schema disclosed in Gorthy shows that the system is able to identify the CLI keywords in XML commands. For example, when presented with the XML command below, the XML schema of Appendix B would validate the XML command by determining whether the XML tags include element names—which are CLI keywords—as defined in the XML schema.

```
<service>
    <alignment>
        <detection/>
    </alignment>
</service>
```

73. Gorthy’s schema identifies XML tags that include an XML parameter to indicate that the XML tag includes one or more CLI keywords because “service,” “alignment,” and “detection” are all valid element names as described and structured in the XML configuration schema. One of ordinary skill in the art would recognize that, in this situation, the “XML parameter to indicate the XML tag includes one or more CLI keywords” is the CLI keyword itself. As discussed above, however, even if a CLI keyword needed to be identified by a separate XML parameter, Froyd illustrates that such an implementation was known in the art long before the ’886 patent and would have been an obvious adaptation of the XML command disclosed by Gorthy.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

74. One of ordinary skill in the art would likewise understand that the process of translating the input XML command into a CLI command involves extracting the one or more CLI keywords from the input command and arranging those CLI keywords into a CLI command according to the CLI syntax of the input command. Gorthy's exemplary schema of Appendix B identifies CLI keywords that would be found in XML tags of an input command in the order in which those CLI keywords would appear in a CLI command. This is evidenced by Appendix A, which shows exemplary CLI commands that could be generated using an XML schema, such as that of Appendix B. Note that the first three commands of Appendix A ("service," "service alignment," and "service alignment detection") include CLI keywords that are sequenced in the same order as those in the schema of Appendix B. Viewing these examples, one of ordinary skill in the art would find it obvious that in translating the input command in XML format (*e.g.*, "<service><alignment></detection></alignment></service>"), Gorthy extracts and arranges the CLI keywords into a CLI command according to the CLI syntax of the input command (*e.g.*, "service alignment detection").

6. [1C.3] "wherein the routing system is configured to perform the operation responsive to the execution of the CLI command;"

75. As discussed with reference to claim 1B.1 above, Gorthy discloses that the routing system is configured to perform the operation responsive to the ex-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

ecution of the CLI command. Gorthy explains that “[o]nce reformatted into a CLI format, the command can be pushed out to the appropriate router. Thus, a system administrator could configure such a router without knowing the specifics of the CLI.” Ex. 1003 at 3:16-19; *see also id.* at 4:33-35 (“[N]ew configuration commands are provided to the router 120 through the CLI.”). Gorthy further explains that a CLI-based command is “passed to the configuration storage module 145 *where it is integrated into the configuration of the router.*” *Id.* at 6:12-15 (emphasis added). One of skill in the art would understand from this disclosure that the routing system is configured to perform the operation responsive to the execution of the CLI command. For example, when a “service alignment detection” CLI command (*see id.* at Appendix A) is passed to the router, one of skill in the art would understand that the router performs the operation responsive to execution of that CLI command (enabling detection of alignment issues).

7. [1D] “**translating an output message, generated in response to performance of the operation, from a CLI format into an XML format having the CLI syntax, wherein the translating includes parsing the output message to identify at least one CLI token, translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values, and generating the output message in the XML format with the XML values;**”

76. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses translating an output message, generated in response to perfor-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

mance of the operation, from a CLI format into an XML format having the CLI syntax, wherein the translating includes parsing the output message to identify at least one CLI token, translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values, and generating the output message in the XML format with the XML values. For instance, Courtney discloses a converter that translates an output message in a CLI format to an XML format. *See* Ex. 1002 at 2:40-45. One of skill in the art would appreciate that one sort of output message that can be translated is an output message providing the current configuration of the system. This can be done through what Gorthy refers to as “a command extraction mode,” which “is activat[ed] by entering a ‘?’ at the prompt.” Ex. 1003 at 4:48-49. The submission of such a command leads to a response, or series of responses, that allow a user to “retrieve[] the primary commands, subcommands and bounds” of a system. *Id.* at 4:50-51; *see also* Ex. 1002 at 4:48-49 (“available commands are returned through the CLI”). Messages output in response to such commands requesting configuration information regarding a system can thereafter be converted into XML format: “In certain embodiments, this schema can be directly used to generate an XML document that represents the configuration of the particular network device.” *Id.* at 3:12-14. When translating from CLI to XML, the obvious and most sensible approach to take would be to retain the same syntax as the output message being

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

translated, thereby creating a message in an XML format with the CLI syntax. Moreover, the schema ensures that translated commands in XML format have proper CLI syntax.

77. As part of the conversion from CLI to XML, Courtney's system analyzes the output message to extract at least one CLI token (*i.e.*, unit of CLI characters in a sequence). Courtney discloses that one of the first steps in a conversion is to "identif[y] each initial command within each configuration line" from the configuration of the device. Ex. 1002 at 7:17-21. Since such initial commands would be CLI tokens, Courtney's system parses the file in order to extract at least one such token.

78. Courtney also discloses translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values. "[U]sing the identified initial command, the XML converter 235 generates a look-up key that is used to index the hash table, locate a hash map object that corresponds to the look-up key and retrieve that hash map object (steps 275 and 280)." *Id.* at 7:27-31. "The hash map object contains schema information regarding the command or value such as whether optional or required data type, etc." *Id.* at 7:31-33. "Finally, using this hash map object, the XML converter 235 can assemble the XML-based command and write it to the corresponding XML document (step 295)." *Id.* at 7:33-36. This process is repeated for each command in

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

the network device's native-format configuration. *Id.* at 7:37-38. One of skill in the art would also understand that the conversion from CLI tokens to XML values inherently requires that some mapping exist between the two and that they be stored, even if only in a transitory manner, to accomplish the desired conversion. The message will thereafter be converted into XML format, using the XML values.

79. The JUNOScript Guide provides another example in the prior art of translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values. The JUNOScript Guide explains that the output from a CLI command is, by default, formatted ASCII text. Ex. 1005 at 37. It provides the following example of an output message produced in response to a CLI command:

Physical interface: **fxp0**, **Enabled**, Physical link is **Up**
Interface index: **4**, SNMP ifIndex: **3**

Id. at 17 (emphasis added).

80. The JUNOScript Guide explains that the JUNOScript API translates this output message from a CLI format into an XML format having a CLI syntax. In doing so, JUNOScript parses the output message shown above to identify at least one CLI token and translates each token to a corresponding XML value, thus generating the output message in XML format with XML values:

```
<interface>
  <name>fxp0</name>
  <admin-status>enabled</admin-status>
```

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

```

        <operational-status>up</operational-status>
        <index>4</index>
        <snmp-index>3</snmp-index>
    </interface>

```

Id. (emphasis added).

81. In another example, the JUNOScript Guide explains that the JUNOScript API translates an output message, generated in response to the performance of an operation, from a CLI format into an XML format having a CLI syntax by using the “pipe” function and the “display xml” command:

```

user@host> show chassis hardware | display xml
<rpc-reply xmlns:junos="http://xml.juniper.net/junos/5.1R1/dtd/junos.dtd">
<chassis-inventory xmlns="http://xml.juniper.net/junos/5.1R1/dtd/junos-chassis.dtd">
<chassis junos:style="inventory">
<name>Chassis</name>
<serial-number>00118</serial-number>
<description>M40</description>
<chassis-module>
<name>Backplane</name>
<version>REV 06</version>
<part-number>710-000073</part-number>
<serial-number>AA2049</serial-number>
</chassis-module>
<chassis-module>
<name>Power Supply A</name>
...
</chassis-module>
...
</chassis>
</chassis-inventory>
</rpc-reply>

```

Id. at 37.

82. The JUNOScript Guide examples show that the output message generated in response to a CLI command is parsed to identify at least one CLI token (e.g., “fxp0,” “enabled,” etc.), and that each CLI token is translated into a corresponding XML value (e.g., “<name>fxp0</name>,” “<admin-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

status>enabled</admin-status>”). The JUNOScript Guide recognizes that, by default, the output of a CLI command to the JUNOScript server will be in CLI format. *Id.* (“To display the output from a JUNOS CLI command as JUNOScript tags rather than the default formatted ASCII, pipe the command to the display xml command.”). By using the “| display xml” feature, the JUNOScript API will translate the default CLI output to XML. *Id.* In doing so, it is both inherent and obvious that the JUNOScript API translates the output message “according to a stored mapping of CLI tokens-to-XML values,” as there would be no other way for the software to produce the XML output from a pipe (“|”) function, which uses the default CLI command output as the input for the “display xml” function. In other words, because we know that a CLI format output message is the input to the “display xml” function, one of ordinary skill in the art would understand that it would be obvious that the JUNOScript API accesses a stored mapping to produce the XML-format output message shown on page 37 of the JUNOScript Guide. The result is an output message in XML format with the XML values, as shown in the above example from page 37 of the JUNOScript Guide.

8. [1E] “and transmitting the output message in the XML format having the CLI syntax to a remote device external from the routing system.”

83. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses transmitting the output message in the XML format having the

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

CLI syntax to a remote device external from the routing system. For instance, Courtney discloses that the output from the XML converter can be passed to relevant software applications, which one of ordinary skill would understand to include software applications running outside the routing system. *See* Ex. 1002 at 6:50-60. This is also confirmed by Courtney's mention of the fact that the configuration for a network device could be obtained from "an alternate location." *Id.* at 2:43. One of ordinary skill would understand that converted XML commands also could be transmitted to other types of remote devices, and that one of the reasons to use an easily transferable language like XML is to facilitate such transfers. For instance, Froyd discloses presenting converted configuration data in XML format "to the user at the workstation." Ex. 1004 at 14:9-12; *see also id.* at 7:24-25 ("The data [is] then sent to a destination (e.g., user, management application, data file, etc.); *id.* at 15:60-63 ("Placing the statistic information into the XML format also allows the statistic information to be used by other devices capable of processing data in the XML format."). The JUNOScript Guide similarly teaches the transmitting the output message in XML format to a remote device external to the routing system, such as a client application. Ex. 1005 at 15 ("Client applications can configure or request information from a router by encoding the request with JUNOScript tags and sending it to the JUNOScript server running on the router. The JUNOScript server directs the request to the appropriate software modules within

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

the router, encodes the response in JUNOScript tags, and returns the result to the client application.”).

B. Claim 2

- 1. “The method of claim 1, wherein the input command is formatted in accordance with an XML schema of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem.”**

84. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses the method of claim 1, wherein the input command is formatted in accordance with an XML schema of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem. For instance, Gorthy discloses the use of Cisco routers to perform the claimed method. Ex. 1003 at 6:52-55 (explaining that “the embodiments of the present invention are generally described with regard to a router and in particular with regard to Cisco™ routers”). As explained above with respect to claim 1, Gorthy discloses using an XML configuration schema in conjunction with a converter to generate CLI commands from XML-based commands. *Id.* at 2:63-3:11. Gorthy teaches that this XML configuration schema can be generated from the CLI commands associated with the target router (including, for example, a Cisco router). *Id.* at 4:39-41. “After the configuration schema has been generated, it is associated with characteristics of the router and stored accordingly.” *Id.* at 5:50-52. In particular, Gorthy ex-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) plains that “the configuration schema might be associated with a Cisco™ router, model 7500, OS version 12.0.” *Id.* at 5:52-54.

85. The term “internetwork operating system (IOS),” as recited in claim 2, is a term coined by Cisco that refers to operating system software that runs on Cisco routers. Therefore, one of ordinary skill would understand that the CLI rules and behaviors of the disclosed XML configuration schema, when designed to be used with Cisco routers, would be enforced by a component of the internetwork operating system (IOS) (*i.e.*, a command line interface (CLI) parser subsystem).

C. Claim 3

1. **[3A] “The method of claim 2, wherein the translation of the input command from XML format having a CLI syntax into a CLI command comprises:”**

86. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses the method of claim 2, wherein the translation of the input command from XML format having a CLI syntax into a CLI command comprises the remaining elements of claim 3.

2. **[3B] “parsing the input command to identify an XML command attribute;”**

87. Gorthy in combination with Froyd discloses parsing the input command to identify an XML command attribute, for the reasons discussed above with respect to claim element 1C. For instance, as discussed above, one of ordinary skill in the art would understand that the process disclosed in Gorthy of translating the

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

input command into a CLI command involves analyzing XML tags to identify keywords and parameters. *See* discussion of claim element 1C. Moreover, in Appendix B, Gorthy provides an example of an XML configuration schema that includes XML tags that contain XML parameters indicating that the tag contains one or more CLI keywords. The XML schema disclosed in Gorthy thus shows that the system is able to identify these things.

88. Moreover, as discussed above, the Froyd patent, which specifically claims converting XML configuration files into CLI configuration commands, specifically discloses that an XML command in CLI syntax can be converted to a CLI command. *See* Ex. 1004 at 17:57-60 (claim 5); *see also id.* at 17:51-52 (claim 2). For instance, in the course of discussing Figure 4 (a table illustrating exemplary XML tags), Froyd explains that “[t]he <command> tag 415 is the top-level tag for a single command.” *Id.* at 8:51-52. Froyd further explains that “[t]he <command> tag 415 contains a single <keyword> tag” and that “[t]he <keyword> tag 420 specifies the text of a command keyword in the command set of the CLI such as, for example SHOW, ENABLE or LOGOUT.” *Id.* at 8:53-65. “The text is supplied by the single attribute ‘text’.” *Id.* at 8:65-67. Through this example, Froyd provides an illustration of an XML tag with an XML parameter (“text”) indicating that the XML tag includes one or more CLI keywords. One of ordinary skill would under-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

stand that, as part of converting from XML to CLI, the command would be parsed to identify the XML command attribute.

3. [3C] “traversing the input after the XML command attribute to identify any keywords and any parameters associated with the XML command attribute;”

89. Gorthy in combination with Froyd discloses traversing the input after the XML command attribute to identify any keywords and any parameters associated with the XML command attribute. As discussed above with respect to claim 1, for instance, one of ordinary skill would understand that the process disclosed in Gorthy of translating the input XML command into a CLI command involves identifying keywords and parameters. In Appendix B, for instance, Gorthy provides an example of an XML configuration schema that includes XML tags that contain XML parameters indicating that the tag contains one or more CLI keywords. The XML schema disclosed in Gorthy thus shows that the system is able to identify these things. One of ordinary skill would understand that the translation process involves traversing the input to identify keywords and parameters associated with the XML tag. *See* discussion of claim element 1C; Ex. 1004 at 8:51-67.

4. [3D] “translating the XML command attribute into the CLI command;”

90. Gorthy in combination with Froyd discloses translating the XML command attribute into the CLI command. *See* discussion of claim element 1C. As discussed above, Gorthy discloses a system that translates XML-based configura-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

tion commands having CLI syntax to CLI-based commands using an XML schema. *See* discussion of claim element 1C; Ex. 1003 at 3:9-16. The XML configuration schema is used to “reformat the XML-based command into a proper CLI format.” Ex. 1003 at 3:15-16. One of ordinary skill in the art would understand that this conversion process involves translating XML tags containing XML parameters and CLI keywords into CLI commands. *See* discussion of claim element 1C; Ex. 1004 at 8:51-67.

5. [3E] “and translating the keywords and any parameters into associated attributes of the CLI command.”

91. Gorthy in combination with Froyd discloses translating the keywords and any parameters into associated attributes of the CLI command. *See* discussion of claim element 1C. As discussed above, Gorthy discloses a system that translates XML-based configuration commands having CLI syntax to CLI-based commands using an XML schema. *See* Ex. 1003 at 3:9-16. The XML configuration schema is used to “reformat the XML-based command into a proper CLI format.” *Id.* at 3:15-16. One of ordinary skill in the art would understand that this conversion process involves identifying keywords and any parameters contained in XML tags, and then translating those keywords and parameters into associated attributes of the CLI commands. *See* discussion of claim element 1C; Ex. 1004 at 8:51-67.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

D. Claim 4

- 1. “The method of claim 1, wherein the output message in the XML format having the CLI syntax includes data formatted in accordance with an XML data model of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem.”**

92. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses the method of claim 1, wherein the output message in the XML format having the CLI syntax includes data formatted in accordance with an XML data model of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem. As discussed above with respect to claim element 1D, Courtney discloses a system that translates an output message in a CLI format into an XML format. *See* discussion of claim element 1D; Ex. 1002 at 2:40-49; *see also* Ex. 1003 at 4:48-51. For instance, Courtney discloses a schema that “can be directly used to generate an XML document that represents the configuration of the particular network device.” Ex. 1002 at 3:12-14. This “schema can include a standard representation of the command structure for a particular type of network device” (*id.* at 3:1-3), and Courtney specifically discloses that Cisco routers may be used. Courtney teaches, for instance, that “one schema could contain a representation of the command structure for all model 7500 Cisco™ routers using OS version 12.1, and another schema could contain a representation of the command structure routers using OS version 12.2.” *Id.*

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) at 3:3-7. This schema ensures that translated commands in XML format will be formatted in accordance with proper CLI rules and behaviors.

93. Moreover, as noted above with respect to claim 2, the term “internet-work operating system (IOS)” is a term coined by Cisco that refers to operating system software that runs on Cisco routers. Therefore, one of ordinary skill would understand that the CLI rules and behaviors, when designed to be used with Cisco routers, are enforced by a component of the internetwork operating system (IOS) (*i.e.*, a command line interface (CLI) parser subsystem).

E. Claim 5

1. **[5A] “The method of claim 1, wherein the translation of the output message from the CLI format into the XML format having the CLI syntax comprises:”**

94. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses the method of claim 1, wherein the translation of the output message from the CLI format into the XML format having the CLI syntax comprises the remaining elements of claim 5.

2. **[5B] “parsing the output message to identify at least one CLI token;”**

95. This element is identical to the corresponding requirement in claim element 1D. Thus, Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses parsing the output message to identify at least one CLI token for the reasons set forth above. *See* discussion of claim element 1D. For in-

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

stance, Courtney discloses that one of the first steps in a conversion is to “identif[y] each initial command within each configuration line” from the configuration of the device. Ex. 1002 at 7:17-21. Since such initial commands would be CLI tokens, Courtney’s system parses the file in order to identify at least one such CLI token.

3. [5C] “accessing a stored mapping of CLI tokens-to-XML values;”

96. As discussed above with respect to claim element 1D, Courtney discloses the use of a “schema [that] can be directly used to generate an XML document that represents the configuration of the particular network device.” *Id.* at 3:12-14. One of skill in the art would understand that when converting between CLI tokens and XML values, the mapping between the two must be stored, even if only in a transitory manner, to accomplish the desired conversion. One of ordinary skill would also appreciate that, in order to perform the desired conversion, the system would access the stored mapping of CLI tokens-to-XML values. *See* discussion of claim element 1D. One of ordinary skill in the art would recognize that the JUNOScript API, for example, must access a mapping of CLI tokens-to-XML values when the API is instructed to output the results of a CLI command in XML, such as by using the “| display xml” command, rather than the default ASCII-formatted text. *See* Ex. 1005 at 37; *see also id.* at 17.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

4. [5D] “translating each CLI token of the output message into a corresponding XML value, in accordance with said stored mapping;”

97. This element is substantively identical to the corresponding requirement in claim element 1D of “translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values.” As discussed above, Courtney discloses the use of a “schema [that] can be directly used to generate an XML document that represents the configuration of the particular network device.” Ex. 1002 at 3:12-14. One of skill in the art would understand that when converting between CLI tokens and XML values, the mapping between the two must be stored, even if only in a transitory manner, to accomplish the desired conversion. *See* discussion of claim element 1D. One of ordinary skill in the art would recognize that the JUNOScript API, for example, translates the output message of a CLI command according to stored mapping of CLI tokens-to-XML values when using the “display xml” command. *See* discussion of claim element 1D; *see also* Ex. 1005 at 17, 37.

5. [5E] “and generating the output message in the XML format having the CLI syntax with the XML values.”

98. This element is substantially identical to the corresponding requirement in claim element 1D. Thus, Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses generating the output message in the XML format having the CLI syntax with the XML values, for the reasons set forth above. *See*

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

discussion of claim element 1D. For instance, Courtney discloses a converter that translates an output message in a CLI format to an XML format (*see* Ex. 1002 at 2:40-45), and, in view of the JUNOScript Guide, one of skill in the art would appreciate that one sort of output message that can be translated is an output message providing the current configuration of the system. *See* Ex. 1003 at 4:48-51; *see also* Ex. 1002 at 4:48-49; Ex. 1005 at 37. Messages output in response to such commands requesting configuration information regarding a system can thereafter be converted into XML format using a schema. Ex. 1002 at 3:12-14 (“In certain embodiments, this schema can be directly used to generate an XML document that represents the configuration of the particular network device.”). As discussed above, when translating from CLI to XML, the most obvious and sensible approach would be to retain the same syntax as the output message being translated, thereby creating a message in an XML format with the CLI syntax. Indeed, the output messages illustrated in the JUNOScript Guide reflect this approach (*see, e.g.,* Ex. 1005 at 17, 37) and would only further confirm to one of ordinary skill in the art that this was the most obvious solution. *See* discussion of claim element 1D.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

F. Claim 6

- 1. [6A] “A computer-usable memory device having computer-readable program code embedded therein for causing a computer system to:”**

99. Claim 6 is substantially identical to claim 1, except that it is drafted as a computer-usable memory device for causing a system to perform a recited method. Because the claimed method to be performed by this computer-usable memory device is substantially the same as claim 1, this claim is invalid for the reasons articulated in claim 1 above, and I incorporate that discussion by reference. My analysis below addresses the minor variations between this claim and claim 1.

100. Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses a computer-usable memory device having computer readable program code embedded therein for causing a system to perform the steps set forth in the elements below. For example, the invention of Gorthy is claimed as a computer-readable storage medium (*i.e.*, memory device) with program instructions (*i.e.*, computer-readable program code) stored on it for performing the claimed method. Ex. 1003 at 12:4-26. Moreover, Froyd notes that its invention “may be implemented by a processing unit in a digital processing system, which executes sequences of computer program instructions which are stored in a memory which may be considered to be a machine-readable storage media.” Ex. 1004 at 16:49-54. Moreover, Froyd explains that the memory device “may be random access memory, read only

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) memory, a persistent storage memory, such as mass storage device or any combination of these devices.” *Id.* at 16:54-57. And Froyd confirms that when the code is executed, it “causes the processing unit to perform operations according to” the invention. *Id.* at 16:57-59; *see also id.* at 18:9-67 (claims 11-20). Likewise, Courtney discloses that its invention can be carried out using “a plurality of instruction stored on [a] storage device,” where “the plurality of instructions [are] configured to cause the processor” to carry out the steps of the invention. Ex. 1002 at 10:3-5 (claim 12).

2. **[6B] “receive an input command requesting an operation be performed by a routing system, wherein the input command is configured in an extensible markup language (XML) format having command a line interface (CLI) syntax with CLI keywords sequenced according to configuration rules for CLI commands;”**

101. This limitation is substantially identical to that of claim 1B, and my above analysis of that claim accordingly applies to this limitation. *See* discussion of claims 1B.1 and 1B.2. Claim 6B lacks the requirement of claim 1B.1 that the input command be received “with a command line interface (CLI) parser,” and my analysis of the narrower claim 1B.1 limitation thus applies here. Claim 6B requires that the input command “request an operation be performed by a routing system,” whereas claim 1B.1 requires the input command “be configured to request an operation be performed by a routing system.” This distinction does not affect my analysis of claim 1B.1 and 1B.2, as the XML configuration command of Gorthy is both

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

“configured to request” that an operation be performed by a routing system and, moreover, that the command actually “requests” the operation be performed. *See* Ex. 1003 at 3:9-19, 6:3-15. For these and the reasons discussed above as to Claims 1B.1 and 1B.2, Gorthy discloses this limitation.

3. **[6C] “translate the input command from the XML format having the CLI syntax into a CLI command, wherein the routing system is configured to execute the CLI command and perform the operation, and wherein the computer system is further configured to translate the input command by identifying at least one XML tag that includes an XML parameter to indicate the XML tag includes one or more CLI keywords, extracting the one or more CLI keywords from the input command, and arranging the one or more CLI keywords into the CLI command according to the CLI syntax of the input command;”**

102. This limitation is substantially identical to that of claim 1C, and my above analysis of that claim accordingly applies to this limitation. *See* discussion of claims 1C.1, 1C.2, 1C.3. Claim 6C lacks the requirement of claim 1C.1 that the input command be translated “with the CLI parser,” and my analysis of the narrower claim 1C.1 limitation thus applies here. Claim 6C requires that the routing system be “configured to execute the CLI command and perform the operation,” whereas claim 1C.2 requires that the CLI command “when executed, is configured to prompt the routing system to perform the operation” and that the “routing system is configured to perform the operation responsive to the execution of the CLI command.” My analysis of claim 1C.2 applies to claim 6C because, in both claims,

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

Gorthy's input command, once converted to a CLI command, is executed by the routing system, which causes the routing system to perform the operation. *See* Ex. 1003 at 3:9-19, 6:3-15, 6:21-26; *see also id.* at Fig. 8. Whether claimed as the input command that prompts an operation to be performed when executed or the routing system that executes the command, prompting the operation to be performed, Gorthy discloses each limitation. For these and the reasons discussed above as to claims 1C, Gorthy discloses this limitation.

4. **[6D] “translate an output message from a CLI format into XML format having the CLI syntax, wherein the output message is generated in the CLI format by the routing system responsive to the performance of the operation, and wherein the translating includes parsing the output message to identify at least one CLI token, translating each CLI token of the output message into a corresponding XML value according to a stored mapping of CLI tokens-to-XML values, and generating the output message in the XML format with the XML values;”**

103. This limitation is substantially identical to that of claim 1D, and my above analysis of that claim accordingly applies to this limitation. *See* discussion of claim 1D. The difference between these limitations is that claim 1D specifies that the output message is “generated in response to performance of the operation,” whereas claim 6D indicates the output message is “generated in the CLI format by the routing system responsive to the performance of the operation.” My analysis of claim 1D applies because the output message of Courtney is generated by the routing system (*e.g.*, a Cisco router) in CLI format in response to the operation. *See* Ex.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) 1002 at 4:48-49; *see also* Ex. 4:48-51. Likewise, the output message generated by the JUNOScript API, running on a Juniper Networks router, is natively in a CLI format in response to performing the operation. Ex. 1005 at 37. For these and the reasons discussed above as to claim 1D, Gorthy discloses this limitation.

5. [6E] “and transmit the output message in the XML format having the CLI syntax to a remote device external from the routing system.”

104. This limitation is identical to that of claim 1E, except that claim 1E uses the gerund, “transmitting,” rather than claim 6E’s infinitive “[to] transmit.” Accordingly, my above analysis of claim 1E applies to this limitation. *See* discussion of claim 1E. For the reasons identified above as to claim 1E, Gorthy in combination with Froyd discloses this limitation.

G. Claim 7

1. “The computer-usable memory device of claim 6, wherein the input command is formatted in accordance with an XML schema of CLI rules and behaviors enforced by an internetwork operating system (IOS) command line interface (CLI) parser subsystem.”

105. Claim 7 recites the same method as claim 2, but is drafted in the form of a computer-usable memory device with embedded code for performing the claimed steps. As noted above with respect to claim element 6A, both Courtney and Froyd expressly disclose such a computer-usable memory device. *See* Ex. 1004 at 16:49-59; *see also id.* at 18:9-67 (claims 11-20); Ex. 1002 at 10:3-5 (claim 12). Moreover, one of ordinary skill in the art would recognize that an obvious and

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

conventional way to allow a method to be practiced is to place code that would carry out the method onto a computer-usable memory device.

106. Because the method performed using the computer-readable memory device of claim 7 is identical to that recited in claim 2, this claim is disclosed by Gorthy in combination with Courtney and Froyd as discussed above. For instance, as explained with respect to claim 2, Gorthy discloses the use of Cisco routers to perform the claimed method. Ex. 1003 at 6:52-55 (explaining that “the embodiments of the present invention are generally described with regard to a router and in particular with regard to Cisco™ routers”); *see also id.* at 4:39-41, 5:50-54. The term “internetwork operating system (IOS)” is a term coined by Cisco that refers to operating system software that runs on Cisco routers. Therefore, one of ordinary skill would understand that the CLI rules and behaviors of the XML configuration schema, disclosed in Gorthy (*see id.* at 4:39-41), when designed to be used with Cisco routers, are enforced by a component of the internetwork operating system (IOS) (*i.e.*, a command line interface (CLI) parser subsystem).

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

H. Claim 8

1. [8A] **“The computer-usable memory device of claim 7, wherein the translation of the input command from XML format having a CLI syntax into a CLI command comprises:”**

107. Claim 8 recites the same method as claim 3, but is drafted in the form of a computer-usable memory device with embedded code for performing the claimed steps. As noted above with respect to claim element 6A, both Courtney and Froyd expressly disclose such a computer-usable memory device. *See* Ex. 1004 at 16:49-59; *see also id.* at 18:9-67 (claims 11-20); Ex. 1002 at 10:3-5 (claim 12). Moreover, one of ordinary skill in the art would recognize that an obvious and conventional way to allow a method to be practiced is to place code that would carry out the method onto a computer-usable memory device. Because the method performed using the computer-readable memory device of claim 8 is identical to that recited in claim 3, this claim is disclosed by Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide as discussed above. *See* discussion of claim 3.

2. [8B] **“parsing the input command to identify an XML command attribute;”**

108. Gorthy in combination with Courtney and Froyd discloses this claim limitation for the reasons discussed above with respect to claim elements 1C and 3B. For instance, as explained above, one of ordinary skill in the art would understand that the process disclosed in Gorthy of translating the input command into a

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

CLI command involves analyzing XML tags to identify keywords and parameters. *See* discussion of claim element 1C. Moreover, by providing an example in Appendix B of an XML configuration schema that includes XML tags that contain XML parameters indicating that the tag contains one or more CLI keywords, Gorthy thus shows that the system is able to identify these things.

109. In addition, as discussed above, the Froyd patent specifically discloses that an XML command in CLI syntax can be converted to a CLI command. *See* Ex. 1004 at 17:57-60 (claim 5); *see also id.* at 17:51-52 (claim 2). For instance, in the course of discussing Figure 4 (a table illustrating exemplary XML tags), Froyd explains that “[t]he <command> tag 415 is the top-level tag for a single command.” *Id.* at 8:51-52. Froyd further explains that “[t]he <command> tag 415 contains a single <keyword> tag” and that “[t]he <keyword> tag 420 specifies the text of a command keyword in the command set of the CLI such as, for example SHOW, ENABLE or LOGOUT.” *Id.* at 8:53-65. “The text is supplied by the single attribute ‘text’.” *Id.* at 8:65-66. Through this example, Froyd provides an illustration of an XML tag with an XML parameter (“text”) indicating that the XML tag includes one or more CLI keywords. One of ordinary skill would understand that, as part of converting from XML to CLI, the command would be parsed to identify the XML tag.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

3. [8C] “traversing the input after the XML command attribute to identify any keywords and any parameters associated with the XML command attribute;”

110. Gorthy in combination with Courtney and Froyd discloses this claim limitation for the reasons discussed above with respect to claim elements 1C and 3C. As discussed above, one of ordinary skill would understand that the process disclosed in Gorthy of translating the input XML command into a CLI command involves identifying keywords and parameters. In Appendix B, for instance, Gorthy provides an example of an XML configuration schema that includes XML tags that contain XML parameters indicating that the tag contains one or more CLI keywords. The XML schema disclosed in Gorthy thus shows that the system is able to identify these things. One of ordinary skill would understand that the translation process involves traversing the input to identify keywords and parameters associated with the XML tag. *See also* discussion of claim element 1C; Ex. 1004 at 8:51-66.

4. [8D] “translating the XML command attribute into the CLI command;”

111. Gorthy in combination with Courtney and Froyd discloses this claim limitation for the reasons discussed above with respect to claim elements 1C and 3D. As discussed above, for instance, Gorthy discloses a system that translates XML-based configuration commands having CLI syntax to CLI-based commands using an XML schema. *See* discussion of claim element 1C; Ex. 1003 at 3:9-16.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

The XML configuration schema is used to “reformat the XML-based command into a proper CLI format.” *Id.* at 3:15-16. One of ordinary skill in the art would understand that this conversion process involves translating XML tags containing XML parameters and CLI keywords into CLI commands. *See* discussion of claim element 1C; Ex. 1004 at 8:51-67.

5. [8E] “and translating the keywords and any parameters into associated attributes of the CLI command.”

112. Gorthy in combination with Courtney and Froyd discloses this claim limitation for the reasons discussed above with respect to claim elements 1C and 3E. As discussed above, for instance, Gorthy discloses a system that translates XML-based configuration commands having CLI syntax to CLI-based commands using an XML schema. *See* discussion of claim element 1C; Ex. 1003 at 3:9-16. The XML configuration schema is used to “reformat the XML-based command into a proper CLI format.” *Id.* at 3:15-16. One of ordinary skill in the art would understand that this conversion process involves identifying keywords and any parameters contained in XML tags, and then translating those keywords and parameters into associated attributes of the CLI commands. *See* discussion of claim element 1C; Ex. 1004 at 8:51-66.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

I. Claim 9

1. **“The computer-usable memory device of claim 6, wherein the output message in the XML format having the CLI syntax includes data formatted in accordance with an XML data model of CLI rules and behaviors enforced by an inter-network operating system (IOS) command line interface (CLI) parser subsystem.”**

113. Claim 9 recites the same method as claim 4, but is drafted in the form of a computer-usable memory device with embedded code for performing the claimed steps. As noted above with respect to claim element 6A, both Courtney and Froyd expressly disclose such a computer-usable memory device. *See* Ex. 1004 at 16:49-59; *see also id.* at 18:9-67 (claims 11-20); Ex. 1002 at 10:3-5 (claim 12). Moreover, one of ordinary skill in the art would recognize that an obvious and conventional way to allow a method to be practiced is to place code that would carry out the method onto a computer-usable memory device.

114. Because the method performed using the computer-readable memory device of claim 9 is identical to that recited in claim 4, this claim is disclosed by Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide as discussed above. For instance, as explained with respect to claim 4, Courtney discloses a schema that “can be directly used to generate an XML document that represents the configuration of the particular network device.” Ex. 1002 at 3:12-14. This “schema can include a standard representation of the command structure for a particular type of network device” (*id.* at 3:1-3), and Courtney specifically discloses

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

that Cisco routers may be used. *Id.* at 3:3-7. Because Courtney's system is described with regard to Cisco routers, one of ordinary skill would understand that the CLI rules and behaviors, when designed to be used with Cisco routers, are enforced by a component of the internetwork operating system (IOS) (*i.e.*, a command line interface (CLI) parser subsystem).

J. Claim 10

1. **[10A] "The computer-usable memory device of claim 6, wherein the translation of the output message from the CLI format into the XML format having the CLI syntax comprises:"**

115. Claim 10 recites the same method as claim 5, but is drafted in the form of a computer-usable memory device with embedded code for performing the claimed steps. As noted above with respect to claim element 6A, both Gorthy and Froyd disclose such a computer-usable memory device. *See* Ex. 1004 at 16:49-59; *see also id.* at 18:9-67 (claims 11-20); Ex. 1002 at 10:3-5 (claim 12). Moreover, one of ordinary skill in the art would recognize that an obvious and conventional way to allow a method to be practiced is to place code that would carry out the method onto a computer-usable memory device. Because the method performed using the computer-readable memory device of claim 10 is identical to that recited in claim 5, this claim is disclosed by Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide as discussed above.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

2. [10B] “parsing the output message to identify at least one CLI token;”

116. This element is identical to the corresponding requirement in claim elements 1D and 5B. Thus, Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses parsing the output message to identify at least one CLI token for the reasons set forth above. *See* discussion of claim elements 1D and 5C. For instance, Courtney discloses that one of the first steps in a conversion is to “identif[y] each initial command within each configuration line” from the configuration of the device. Ex. 1002 at 7:17-21. Since such initial commands would be CLI tokens, Courtney’s system parses the file in order to identify at least one such CLI token.

3. [10C] “accessing a stored mapping of CLI tokens-to-XML values;”

117. This element is identical to the corresponding requirement in claim element 5C. Thus, Gorthy in combination with Courtney, Froyd, and the JUNOScript Guide discloses accessing a stored map of CLI tokens-to-XML values, for the reasons discussed above. *See* discussion of claim elements 1D and 5C. As discussed above, for instance, Courtney discloses the use of a “schema [that] can be directly used to generate an XML document that represents the configuration of the particular network device.” Ex. 1002 at 3:12-14. When converting between CLI tokens and XML values, the mapping between the two must be stored, even if

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

only in a transitory manner, to accomplish the desired conversion. One of ordinary skill would appreciate that, in order to perform the desired conversion, the system would access the stored mapping of CLI tokens-to-XML values. *See* discussion of claim elements 1D. One of ordinary skill in the art would recognize that the JUNOScript API, for example, must access a mapping of CLI tokens-to-XML values when the API is instructed to output the results of a CLI command in XML, such as by using the “| display xml” command, rather than the default ASCII-formatted text. *See* Ex. 1005 at 37; *see also id.* at 17.

4. [10D] “translating each CLI token of the output message into a corresponding XML value, in accordance with said storage mapping;”

118. This element is identical to the corresponding requirement in claim element 5D. Thus, Gorthy in combination with Courtney, and Froyd, and the JUNOScript Guide discloses translating each CLI token of the output message into a corresponding XML value, in accordance with said stored mapping, for the reasons discussed above. *See* discussion of claim elements 1D and 5D. As discussed above, for instance, Courtney discloses the use of a “schema [that] can be directly used to generate an XML document that represents the configuration of the particular network device.” Ex. 1002 at 3:12-14. One of skill in the art would understand that when converting between CLI tokens and XML values, the mapping between the two must be stored, even if only in a transitory manner, to accomplish the desired

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018) conversion. *See* discussion of claim element 1D. One of ordinary skill in the art would recognize that the JUNOScript API, for example, translates the output message of a CLI command according to stored mapping of CLI tokens-to-XML values when using the “| display xml” command. *See* Ex. 1005 at 37; *see also id.* at 17.

5. [10E] “and generating the output message in the XML format having the CLI syntax with the XML values.”

119. This element is identical to the corresponding requirement in claim elements 1D and 5E. Thus, Gorthy in combination with Courtney, Froyd, and the JUNOScript discloses generating the output message in the XML format having the CLI syntax with the XML values, for the reasons set forth above. *See* discussion of claim elements 1D and 5E. For instance, Courtney discloses a converter that translates an output message in a CLI format to an XML format (*see* Ex. 1002 at 2:37-45), and, in view of the JUNOScript Guide, one of skill in the art would appreciate that one sort of output message that can be translated is an output message providing the current configuration of the system. *See* Ex. 1003 at 4:48-51; *see also* Ex. 1002 at 4:48-49; Ex. 1005 at 37. Messages output in response to such commands requesting configuration information regarding a system can thereafter be converted into XML format using a schema. Ex. 1002 at 3:12-14 (“In certain embodiments, this schema can be directly used to generate an XML document that represents the configuration of the particular network device.”). As discussed

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

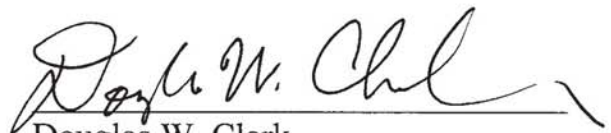
above, when translating from CLI to XML, the most obvious and sensible approach would be to retain the same syntax as the output message being translated, thereby creating a message in an XML format with the CLI syntax. Indeed, the output messages illustrated in the JUNOScript Guide reflect this approach (*see, e.g.*, Ex. 1005 at 17, 37) and would only further confirm to one of ordinary skill in the art that this was the most obvious solution. *See* discussion of claim element 1D.

Inter Partes Review of U.S. Patent 7,953,886 - Declaration of Dr. Clark (Ex. 1018)

120. In signing this declaration, I recognize that the declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required of me, I will appear for cross examination within the United States during the time allotted for cross-examination.

121. I hereby declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct, and that all statements made of my own knowledge are true and that all statements made on information and belief are believed to be true. I understand that willful false statements and the like are punishable by fine or imprisonment or both under 18 U.S.C. § 1001.

Executed on November 24, 2015


Douglas W. Clark